Compilation

JACKSON SOFT

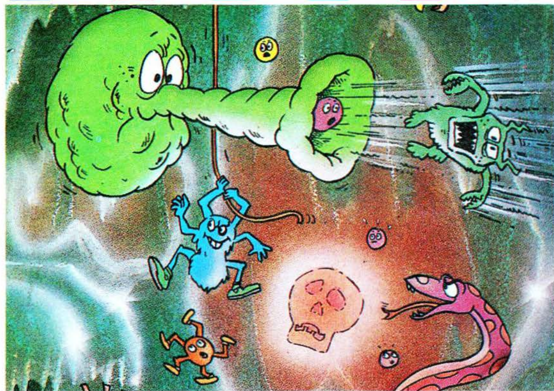
SLURPY

QUAK ATTAK

MLX

SUPERBASIC

BOMBARDIERE



RACCOLTA DI GIOCHI E DI UTILITIES PER

COMMODORE 64



GRUPPO EDITORIALE JACKSON s.r.l.

DIREZIONE, REDAZIONE E AMMINISTRAZIONE
Via Rosellini, 12 - 20124 Milano
Telefoni: 68.03.68 - 68.00.54
68.80.951-2-3-4-5
Telex 333436 GEJ IT
SEDE LEGALE: Via G. Pozzone, 55 - 20121 Milano

DIRETTORE RESPONSABILE:
Giampietro Zanga

COORDINAMENTO EDITORIALE:
A. Cattaneo
P. Todorovich

GRAFICA E IMPAGINAZIONE:
Gianfranco de Rienzo

FOTOCOMPOSIZIONE:
Graphotek
Via Astesani, 16 - Milano
Tel. 64.80.397

STAMPA:
Grafika 78 - Pioltello - MI

AUTORIZZAZIONE ALLA PUBBLICAZIONE:
Trib. di Milano n. 417 del 22-9-'84

PUBBLICITÀ
Concessionario per l'Italia e l'Estero
J. Advertising s.r.l.
V.le Restelli, 5
20124 MILANO
Tel. (02)
68.82.895-68.80.606-68.87.233
Tlx 316213 REINA I
Concessionario esclusivo per la DIFFUSIONE in Italia e Estero:
SODIP - Via Zuretti, 25
20125 MILANO
Spedizione in abbonamento postale Gruppo II/70
Prezzo della rivista L. 6.500
Numero arretrati L. 13.000

Con il presente fascicolo ha inizio una raccolta dei migliori programmi del CBM 64 pubblicati su libri e riviste Jackson.
In ogni numero un supergame originale inglese descritto nei minimi particolari ed in più una miriade di altri programmi scelti tra utility, grafica, musica, giochi ecc.
Ciascun listato è accompagnato da una recensione approfondita e da consigli per un corretto utilizzo.
Non solo potrete caricare immediatamente i programmi grazie al master su cassetta allegato alla confezione, ma anche personalizzarli servendovi dei listati e della descrizione pubblicata all'interno del fascicolo.
La Jackson Soft Compilation, è una pubblicazione creata su misura per voi appassionati allo scopo di arricchire sempre di più la vostra raccolta di programmi.

SOMMARIO

SLURPY

È questo un gioco che vi permetterà di passare qualche serata con gli amici in piacevole competizione.
Mostrate le vostre capacità nel guidare Slurpy, una strana creatura di enorme appetito atterrata per caso sul pianeta Gluton.
Questa creatura ha una bocca a tromba con cui inghiotte qualsiasi cosa gli capiti vicino ed è alla ricerca di cibo nelle popolatissime caverne del pianeta.
Lo scopo del gioco è di aiutare Slurpy a divorare quanto più possibile nella progressiva avanzata all'interno delle caverne di Gluton evitando bocconi avvelenati e il contatto con nemici come Widowmaker, il serpente, l'occhio del diavolo ecc.
Purtroppo Slurpy è invulnerabile solo nella bocca, e nel resto del corpo è estremamente fragile al punto che il minimo contatto con ogni abitante della grotta (eccetto le piccole bolle) provoca la sua morte.
All'inizio del gioco si seleziona il numero dei giocatori, spostando il joystick verso l'alto (1), verso il basso (2) ed il livello di fifficoltà muovendo la cloche nella direzione più adatta alla vostra preparazione (1 su, 3 giù, 2 sinistra, 4 destra).
Premendo il tasto "I" si ha un breve riassunto delle istruzioni con l'indicazione delle creature che potranno essere divorate e altre sempre mortali.
Altri comandi utili sono: "RUN/STOP" che ferma momentaneamente il gioco fintantoché non è premuto nuovamente, e "RESTORE" che vi fa ritornare al menù.
Il cibo principale di Slurpy è costituito dai globuli blu che si muovono all'interno della grotta, quando tutti i globuli sono stati ingoiati. Slurpy passa alla caverna successiva.
La sequenza vitale fa si che i globuli cambino spesso colore diventando rossi e anche color oro.
I globuli rossi non sono commestibili e fanno morire Slurpy se non vengono rigettati immediatamente, i globuli oro fanno diventare Slurpy color oro dandogli maggior forza e rendendolo invincibile al contatto con uno dei nemici.
Ad eccezione dei globuli, Slurpy deve evitare contatti con le altre creature in parti del corpo diverse dalla bocca altrimenti muore.
La bocca di Slurpy risucchia tutte le creature senza distinzioni, sta' a voi reagi-



re sul tasto di sparo del joystick se il boccone appena ingoiato deve essere rigettato perché indigesto oppure no.
Sempre con il tasto di sparo Slurpy cambia direzione alla sua bocca famelica, con la cloche potete invece controllarne la posizione o la direzione nel movimento.
Nel bordo inferiore della caverna vi sono delle uova che si rompono ogni volta che si scontrano con un globulo rosso o altre creature, quando ciò accade, immediatamente un essere della caverna accorre per depositare un altro uovo.
Approfittate di queste occa-



CREATIVE
SPARKS

PRESENTS

SLURPY

PRESS I FOR BRIEF INSTRUCTIONS
PRESS FIRE BUTTON FOR OPTIONS

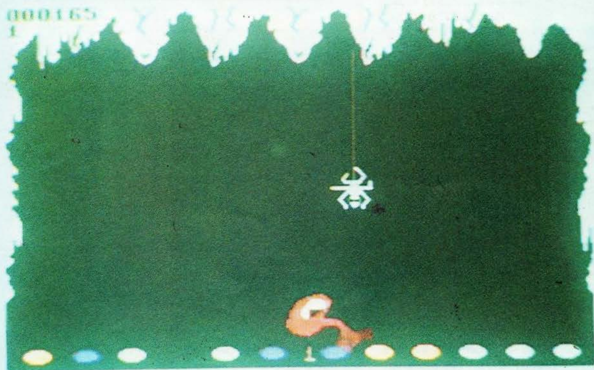
I 1984 BY RANDOM GAMES AND EASYWARE

sioni per divorare a più non posso.
Sul pianeta Gluton sono frequenti terremoti e frane; in tali frangenti prestate attenzione alle rocce che si staccano dal tetto della grotta.
Di caverna in caverna la vita di Slurpy si fa sempre più rischiosa, incontrando sempre più frequentemente creature mortali di tutti i tipi: il teschio rosso le uova gialle che danno vita ad uccelli molto pericolosi ecc.
Il fantasma della caverna che compare ogni tanto è pericoloso quando è visibile ma diventa un boccone prelibato quando scompare.
Come ultima raccomandazione ricordatevi di non divorare più di una creatura alla volta per evitare di intasare l'esofago del vostro Slurpy, dopodiche non ci resta altro che augurarvi un buon divertimento.

Glossario

Press I for brief instructions = Premi "I" per le





istruzioni.
Press fire button for options = Premi fuoco per le opzioni.
Move joystick to select = Selezionare col joystick
Number of players = Numero giocatori
Up = Sopra
Down = Sotto
Player 1 difficulty level = Livello difficoltà giocatore 1
Player 1 press fire to start = Premi fuoco per partire
Press fire button to begin = Premi fuoco per cominciare
Press fire button to flip Slurpy and spit poison (red) objects = Usare il pul-

sante di fuoco per voltare Slurpy o per rigettare gli esseri rossi.

Punteggi

Glowbug = punti 5
Cave bug = punti 20
Cave Bird = punti 30
Serpent = punti 80
Cae Phantom = punti 90
Widowmaker = punti 60
Creeper = punti 70
Deathball, Evil eye = non commestibili.

Comandi

Joystick porta 1 e 2 oppure i seguenti tasti:
In alto = 1
In basso = freccia a sinistra
Sinistra = CTRL
Destra = 2
Fire = Barra

Guida all'input C64

```
TABELLA DI CONVERSIONE

{HOME}.........HOME
{CLR}..........PULIZIA SCHERMO
{CD}...........CURSORE IN BASSO
{CR}...........CURSORE A DESTRA
{CU}...........CURSORE IN ALTO
{CL}...........CURSORE A SINISTRA
{SPC}..........SPAZIO
{RVS ON}.......REVERSE ON
{RVS OFF}......REVERSE OFF
{INST}.........INSERT
{F1}...........TASTO F1
{F2}...........TASTO F2
{F3}...........TASTO F3
{F4}...........TASTO F4
{F5}...........TASTO F5
{F6}...........TASTO F6
{F7}...........TASTO F7
{F8}...........TASTO F8
{WHITE}........COLORE BIANCO
{RED}..........COLORE ROSSO
{GREEN}........COLORE VERDE
{BLUE}.........COLORE BLU
{ORANGE}.......COLORE ARANCIO
{BLACK}........COLORE NERO
{BROWN}........COLORE MARRONE
{LT.GREEN}.....COLORE VERDE CHIARO
{LT.BLUE}......COLORE BLU CHIARO
{PURPLE}.......COLORE PORPORA
{YELLOW}.......COLORE GIALLO
{CYAN}.........COLORE CIANO
{LT.RED}.......COLORE ROSSO CHIARO
{GRAY1}........COLORE GRIGIO 1
{GRAY2}........COLORE GRIGIO 2
{GRAY3}........COLORE GRIGIO 3
```

Norme per la battitura

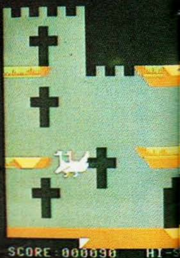
I caratteri grafici, ottenuti con la pressione dei tati "Shift" e "CBM", sono codificati in modo da indicare il tasto da premere assieme a "Shift" o "CBM". Es. il cuoricino è codificato con { SH S }
Il numero dentro le parentesi indica le volte che il tasto va premuto.

QUAK ATTAK



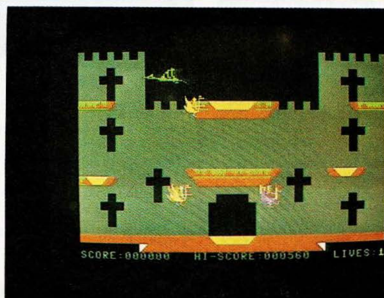
Ecco un gioco di abilità e destrezza ambientato in un leggendario mondo medioevale, dove prodi cavalieri sono in lotta per difendere il loro re e l'onore.
Lo scenario sul quale si svolge l'azione è il vostro castello la cui austera pace è ora violata da perfidi cavalieri rapitori di belle damigelle contro i quali dovete battervi.
Il vostro mandato non ammette discussioni ne voi potete sottrarvi dalla lotta; in sella al vostro fantastico destriero-anatra volante e con la vostra lancia affronterete i nemici senza pietà o per voi sarà la fine.

Dopo aver caricato il programma e premuto F1, compare lo schermo di gioco e i vostri nemici in continuo movimento, con il joystick in porta 1 potete cominciare a studiare il vostro moto che che avviene in modo inusuale.
Con la cloche in avanti vi alzate in volo, in posizione centrale vi abbassate lentamente come planando; premendo il tasto del fuoco lanciate un formidabile colpo di spada.
Contrariamente a quanto potreste immaginare la spada non vi serve per colpire gli attaccanti in volo ma solo per finirli una volta di-



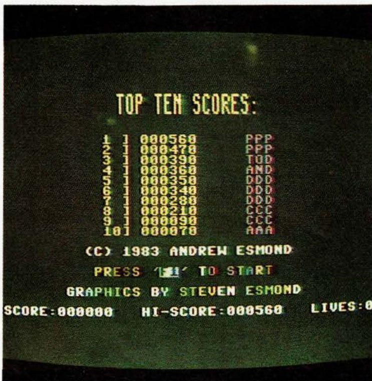

00000 HI-SCORE:000560 LIVES:


RE:000560 LIVES:2


SCORE:000000 HI-SCORE:000560 LIVES:1

sarcionati.
Il combattimento è quindi un continuo tentativo di piombare dall'alto sopra il cavaliere nemico e poi colpirlo senza pietà appena caduto a terra.
A vostra differenza gli attaccanti dispongono di un mezzo di sicurezza inusuale per i tempi, un vero e proprio paracadute grazie al quale possono anche salvarsi dopo il primo attacco; inutile dire che un vostro disarcionamento è sicuramente fatale.
Durante il combattimento dovete prestare attenzione ad un veloce drago volante che attraversa ogni tanto ad altezze diverse lo schermo oltreché all'imprevedibile attacco dei vostri nemici.
Il gioco è organizzato in più livelli nei quali incontrerete avversari sempre più abili e quindi sempre più temibili; la vostra tattica di attacco e difesa dovrà quindi essere studiata volta per volta ricordando sempre che disponete di sole tre vite.
Il punteggio dipende dalla vostra resistenza nonché dall'aver eliminato sistematicamente i cavalieri nemici.
La tavola dei punteggi è la seguente:
disarcionamento: 50 punti
colpi a segno: 20 punti
nemici fuggiti: -50 punti





Comandi (joystick porta 1):
↑= battito ali
< = sinistra
> = destra
fire = colpo di lancia.

Glossario
Top then scores = primi dieci classificati
Scores = punti
Hi-score = punteggio più alto
Lives = vite
Press F1 to start = premi F1 per partire
Please enter your initials = inserisci le tue iniziali
Game over = fine gioco

MLX PER C 64

MLX è una utility che, consentendo il caricamento di programmi in linguaggio macchina praticamente esenti da errori, risparmia tempo e lavoro a chiunque voglia godersi i giochi senza il fastidio di noiosi preliminari.

Se avete già digitato un lungo programma in linguaggio macchina, sarete ben coscienti delle innumerevoli istruzioni DATA seguite da lunghe code di numeri alternati a virgole e del fatto che mai si è sicuri dell'esattezza della copiatura. Così alla fine è sempre necessario tornare indietro, spuntare ogni numero, salvare il programma (attenzione ai "black-out" del LM), lanciarlo, accorgersi che non va a ripetere il ciclo più volte col rischio di rimanere comunque frustrati, o no?

Sino ad ora comunque questo era il miglior modo di caricare programmi in LM in un computer: se non si possiede un assembler e non si ha voglia di districarsi nella programmazione a livello assembly, è mol-

```
100 PRINT"{CLR}{CYAN}";CHR$(142);CHR$(8)
    ;:POKE53281,1:POKE53280,1
101 POKE788,52:REM DISABILITA RUN/STOP
110 PRINT"{RVS ON}{40*SPC}";
120 PRINT"{RVS ON}{15*SPC}{CR}{RVS OFF}{
    CBM *}{SH \}{RVS ON}{CR}{SPC}{CR}{2*
    SPC}{CBM *}{RVS OFF}{CBM *}{SH \}{RV
    S ON}{SH \}{RVS ON}{13*SPC}";
130 PRINT"{RVS ON}{15*SPC}{CR}{CBM N}{CB
    M H}{CR}{SPC}{2*CR}{SPC}{RVS OFF}{SH
     \}{RVS ON}{SH \}{CBM *}{RVS OFF}{CB
    M *}{RVS ON}{13*SPC}";
140 PRINT"{RVS ON}{40*SPC}"
200 PRINT"{2*CD}{PURPLE}{SPC}UN{SPC}PROG
    RAMMA{3*SPC}PER{3*SPC}L'INTRODUZIONE
    {SPC}DI{2*SPC}ROUTINE{SPC}IN{SPC}LIN
    GUAGGIO";
205 PRINT"{SPC}MACCHINA{SPC}A{SPC}PROVA{
    17*SPC}DI{2*SPC}ERRORE{3*CD}"
210 PRINT"{GRAY2}{2*CU}{SPC}INDIRIZZO{SP
    C}DI{SPC}PARTENZA{2*SPC}";:INPUTS:F=
    1-F:C$=CHR$(31+119*F)
220 IFS<256OR(S>40960ANDS<49152)ORS>5324
    7THENGOSUB3000:GOTO210
225 PRINT:PRINT:PRINT
230 PRINT"{GRAY2}{2*CU}{SPC}INDIRIZZO{SP
    C}CONCLUSIVO{3*SPC}";:INPUTE:F=1-F:C
    $=CHR$(31+119*F)
240 IFE<256OR(E>40960ANDE<49152)ORE>5324
    7THENGOSUB3000:GOTO230
250 IFE<STHENPRINTC$;"{RVS ON}INDIRIZZO{
    SPC}CONCLUSIVO<INDIRIZZO{SPC}INIZIAL
    E"
255 IFE<STHENGOSUB1000:GOTO230
260 PRINT:PRINT:PRINT
300 PRINT"{CLR}";CHR$(14):AD=S:POKEV+21,
    0
310 PRINTRIGHT$("0000"+MID$(STR$(AD),2),
    5);":";:FORJ=1TO6
320 GOSUB570:IFN=-1THENJ=J+N:GOTO320
390 IFN=-211THEN710
400 IFN=-204THEN790
410 IFN=-206THENPRINT:INPUT"{CD}{SPC}{SH
     I}NSERIRE{SPC}IL{SPC}{SH N}UOVO{SPC
    }{SH I}NDIRIZZO";ZZ
414 IFN=-206THENIFZZ<SORZZ>ETHENPRINT"{R
    VS ON}ESCE{SPC}DAL{SPC}CAMPO{SPC}DI{
    SPC}VALORI{SPC}INDICATO"
415 IFN=-206THENIFZZ<SORZZ>ETHENGOSUB100
    0:GOTO410
417 IFN=-206THENAD=ZZ:PRINT:GOTO310
420 IFN<>-196THEN480
430 PRINT:INPUT"{SH L}ISTATO:DA";F:PRINT
    "{9*SPC}A";:INPUTT
440 IFF<SORF>EORT<SORT>ETHENPRINT"{SH M}
    INIMO";S;"{SPC}MASSIMO";E;"!{GRAY2}"
```

to più facile digitare un programma BASIC che legga per mezzo di istruzioni DATA i numeri inserendoli in memoria con POKE adeguate.
Alcuni di questi "loaders", così sono conosciuti, controllano le serie di numeri digitati con un checksum. I più semplici sommano tutti i numeri del programma, taluni raggruppano dieci linee o meno; MLX utilizza una speciale lista e li controlla riga per riga, impedendo inoltre di caricare caratteri non previsti (lettere al posto di numeri, ad esempio), oppure numeri superiori a 255 (che non sono ammessi dal LM). MLX arriva persino ad impedire l'immissione di numeri su righe sbagliate: in breve MLX farà diventare obsoleta ogni tecnica di controllo e verifica. Per di più MLX genera un file di uso immediato su nastro o disco e sarà possibile usare il comando LOAD per leggere il programma nel computer.
Più precisamente bisognerà digitare: LOAD "nome programma", 1,1 (nastro)
LOAD "nome programma", 8,1 (floppy).
Per lanciare il programma si userà il comando SYS che trasferisce il controllo dal BASIC al LM. Per prepararsi a caricare tutti i programmi di cui verranno pubblicati solo i listati in LM, occorre digitare il listato di MLX facendo attenzione a non fare errori (purtroppo

```
    :GOTO430
450 FORI=FTOTSTEP6:PRINT:PRINTRIGHT$("00
    00"+MID$(STR$(I),2),5);":";
451 FORK=0TO5:N=PEEK(I+K):PRINTRIGHT$("0
    0"+MID$(STR$(N),2),3);",";
460 GETA$:IFA$>""THENPRINT:PRINT:GOTO310
470 NEXTK:PRINTCHR$(20);:NEXTI:PRINT:PRI
    NT:GOTO310
480 IFN<0THENPRINT:GOTO310
490 A(J)=N:NEXTJ
500 CKSUM=AD-INT(AD/256)*256:FORI=1TO6:C
    KSUM=(CKSUM+A(I))AND255:NEXT
510 PRINTCHR$(18);:GOSUB570:PRINTCHR$(20
    )
515 IFN=CKSUMTHEN530
520 PRINT:PRINT"{RED}{SH L}A{SPC}LINEA{S
    PC}E'{SPC}STATA{SPC}INSERITA{SPC}IN{
    SPC}MANIERA"
525 PRINT"ERRATA.{SPC}{SH R}IPETERE{GRAY
    2}":PRINT:GOSUB1000:GOTO310
530 GOSUB2000
540 FORI=1TO6:POKEAD+I-1,A(I):NEXT:POKE5
    4272,0:POKE54273,0
550 AD=AD+6:IFAD<ETHEN310
560 GOTO710
570 N=0:Z=0
580 PRINT"{CBM +}";
581 GETA$:IFA$=""THEN581
585 PRINTCHR$(20);:A=ASC(A$):IFA=13ORA=4
    4ORA=32THEN670
590 IFA>128THENN=-A:RETURN
600 IFA<>20THEN630
610 GOSUB690:IFI=1ANDT=44THENN=-1:PRINT"
    {CL}{SPC}{CL}";:GOTO690
620 GOTO570
630 IFA<48ORA>57THEN580
640 PRINTA$;:N=N*10+A-48
650 IFN>255THENA=20:GOSUB1000:GOTO600
660 Z=Z+1:IFZ<3THEN580
670 IFZ=0THENGOSUB1000:GOTO570
680 PRINT",";:RETURN
690 S%=PEEK(209)+256*PEEK(210)+PEEK(211)
691 FORI=1TO3:T=PEEK(S%-I)
695 IFT<>44ANDT<>58THENPOKES%-I,32:NEXT
700 PRINTLEFT$("{3*CL}",I-1);:RETURN
710 PRINT"{CLR}{RVS ON}***{SPC}{SH S}AVE
    {SPC}***{3*CD}"
720 INPUT"{CD}{SH N}OME{SPC}DEL{SPC}FILE
    ";F$
730 PRINT:PRINT"{2*CD}{RVS ON}{SH N}{RVS
     OFF}ASTRO{SPC}O{SPC}{RVS ON}{SH D}{
    RVS OFF}ISCO:{SPC}({SH N}/{SH D})"
740 GETA$:IFA$<>"N"ANDA$<>"D"THEN740
750 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
760 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T
    $):POKE782,ZK/256
```

```
762 POKE781,ZK-PEEK(782)*256:POKE780,LEN
    (T$):SYS65469
763 POKE780,1:POKE781,DV:POKE782,1:SYS65
    466
765 POKE254,S/256:POKE253,S-PEEK(254)*25
    6:POKE780,253
766 POKE782,E/256:POKE781,E-PEEK(782)*25
    6:SYS65496
770 IF(PEEK(783)AND1)OR(ST AND191)THEN78
    0
775 PRINT"{CD}{SH F}ATTO.":END
780 PRINT"{CD}{SH E}RRORE{SPC}NEL{SPC}{S
    H S}AVE-RIPROVA!":IFDV=1THEN720
781 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1
    $;E2$:CLOSE15:GOTO720
790 PRINT"{CLR}{RVS ON}***{SPC}{SH L}OAD
    {SPC}***{2*CD}"
800 INPUT"{2*CD}{SH N}OME{SPC}DEL{SPC}FI
    LE";F$
810 PRINT:PRINT"{2*CD}{RVS ON}{SH N}{RVS
     OFF}ASTRO{SPC}O{SPC}{RVS ON}{SH D}{
    RVS OFF}ISCO:{SPC}({SH N}/{SH D})"
820 GETA$:IFA$<>"N"ANDA$<>"D"THEN820
830 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
840 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T
    $):POKE782,ZK/256
841 POKE781,ZK-PEEK(782)*256:POKE780,LEN
    (T$):SYS65469
845 POKE780,1:POKE781,DV:POKE782,1:SYS65
    466
850 POKE780,0:SYS65493
860 IF(PEEK(783)AND1)OR(ST AND191)THEN87
    0
865 PRINT"{CD}{SH F}ATTO.":GOTO310
870 PRINT"{CD}{SH E}RRORE{SPC}NEL{SPC}{S
    H L}OAD-RIPETI!{CD}":IFDV=1THEN800
880 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1
    $;E2$:CLOSE15:GOTO800
1000 REM CICALINO
1001 POKE54296,15:POKE54277,45:POKE54278
     ,165
1002 POKE54276,33:POKE54273,6:POKE54272,
     5
1003 FORT=1TO200:NEXT:POKE54276,32:POKE5
     4273,0:POKE54272,0:RETURN
2000 REM CAMPANELLO
2001 POKE54296,15:POKE54277,0:POKE54278,
     247
2002 POKE54276,17:POKE54273,40:POKE54272
     ,0
2003 FORT=1TO100:NEXT:POKE54276,16:RETUR
     N
3000 PRINTC$;"{RVS ON}{SPC}NON{SPC}IN{SP
     C}PAGINA{SPC}ZERO{SPC}O{SPC}SU{CR}R
     OM{SPC}":GOTO1000
```

non è in grado di correggere anche se stesso). L'uso finale è semplice; lanciando MLX verranno richiesti due numeri: l'indirizzo di partenza e l'indirizzo di fine; questi numeri, indispensabili, saranno dati di volta in volta insieme a quelli da indicare nella SYS.

MLX non usa il normale "screen editor" del computer, accetta solo input numerici; se si rende necessaria una correzione nell'ambito della riga su cui si sta lavorando, il tasto INST/DEL è stato predisposto a cancellare l'intero numero e può essere ripetuto sino al principio della riga.

Ogni tre cifre digitate, il programma provvede a stampare una virgola e si dispone ad accettare il numero successivo; a questo scopo i listati sono composti da numeri di tre cifre. Per omettere gli zeri non significativi sarà necessario invece premere RETURN, la virgola oppure la barra spaziatrice per passare al numero successivo. La somma di verifica appare automaticamente in "reverse"; cosa che non influenza lo svolgimento dell'operazione, ma crea un effetto estetico di completezza.

Nella fase di sperimentazione si è trovato che anche il caricamento di lunghi listing diventa molto più semplice e veloce; avendo a disposizione una cuffia audio e conoscendo bene la tastiera, si potrebbe persino

fare a meno del televisore; in ogni caso neanche le persone più inesperte sono mai riuscite a commettere errori nei testi di MLX.
Una volta caricato in memoria il listing in LM, si dovrà registrarlo su nastro o disco seguendo le istruzioni illustrate sullo schermo.
Se il sistema operativo trova errori nella fase di SAVE, verificare innanzitutto la periferica interessata (attenzione a: fine nastro, disco pieno, ecc.) poi rivedere il programma MLX (ricordarsi che non può verificare se stesso). Se si vuole digitare il listato in più fasi, MLX permette di salvare la parte completata e caricarla in memoria per continuare quando e quante volte sia necessario.
MLX riconosce i seguenti comandi speciali:
SHIFT + S = SAVE
SHIFT + L = LOAD
SHIFT + N = nuovo indirizzo
SHIFT + D = display
ogni volta che si preme una di queste combinazioni di tasti (mantenendo SHIFT premuto) si esce dalla riga in corso di scrittura, pertanto è consigliabile eseguire i comandi a principio di riga. Si usa Save per conservare ciò che si è realizzato sino a quel momento, l'istruzione opererà come di solito sulla periferica prescelta; il programma non sarà però eseguibile fino al suo complemento. È importante, in caso di spezzettamento del lavoro in più parti, ricordare l'indirizzo a cui si era terminata la parte precedente (prenderne nota). Alla ripresa occorrerà lanciare MLX, rispondere alla prima fase con gli stessi indirizzi di start ed end poi, dopo il numero della prima riga usare il comando Load per caricare la parte precedentemente memorizzata; a questo punto per riprendere il lavoro inserire, per mezzo del comando SHIFT + N, l'indirizzo che ci si era annotato; è indispensabile usare sempre un indirizzo tra quelli che fanno da capoverso nel listato altrimenti l'automatismo di verifica segnalerà un errore di somma. Il comando Display corrisponde a LIST, infatti seguito da due indirizzi nella gamma dichiarata esegue una lista dei valori contenuti nelle locazioni richiamate; questa routine si interrompe premendo un qualsiasi tasto.
Questi comandi speciali possono apparire poco utilizzabili ad una prima lettura, in realtà il programma diventa uno dei tool (attrezzi) a disposizione più completi e come tale deve prevedere le varie possibilità di lavoro; come comportarsi ad esempio se alla ripresa di un lavoro ci si accorge di aver dimenticato l'indirizzo da cui partire?
Con il comando Display sarà facile cercare dove il listato si interrompe e ripartire. I comandi Load e Save sono indispensabili per fare più copie del programma completato e provato, cosa altrimenti non realizzabile con i metodi usuali.
Nella routine di SAVE, durante il LOAD, la scritta FOUND "nome programma" può apparire più volte di seguito, ma questo non influenza il regolare caricamento.
Molto interessante è l'uso delle routine del KERNAL di SAVE e LOAD a cui si ha accesso con il comando SYS dopo aver caricato i due indirizzi di partenza e fine locazioni 251/252 (lowbyte/hig byte) e 254/255 rispettivamente; eventuali errori vengono rilevati e posti nella locazione 253 sotto forma di un numero inferiore a 10.
Siamo sicuri di avere proposto con MLX una utility che faccia realmente risparmiare tempo e fatica incoraggiando l'utilizzo dei programmi in LM e permettendo di goderne la qualità.
Ricordatevi di registrare più copie di MLX: vi sarà certo utile per i numerosi programmi, di prossima pubblicazione.

Pubblicato da SuperVic e C.64 n. 2 1984

SUPERBASIC è una utylity che aggiunge nuovi comandi per la gestione di: sprite, colori, grafica in alta risoluzione, musica, organizzazione della memoria, etc.
Questi comandi possono essere usati sia in modo diretto che da programma, e sono formati dal carattere "[" più un codice mnemonico di quattro lettere. L'unico prezzo da pagare è il non utilizzo del comando STOP, che può comunque essere agevolmente sostituito da END. Non diamo il listato in linguaggio macchina per ovvie ragioni di spazio ed anche perché registrato sulla cassetta allegata. Le istruzioni aggiunte dal SUPERBASIC possono essere suddivise in sette categorie principali:

- standard modificate
- per il controllo dei suoni
- per il controllo dei colori
- per il controllo degli sprite
- per il controllo del chip VIC II
- per l'uso del banco di memoria 3
- per il controllo della grafica hi-res

Mai più quindi lunghe serie di POKE per ottenere un suono od un lentissimo plottaggio di una curva in pagina grafica! Basta un comando e si ottiene ciò che si vuole con la velocità del linguaggio macchina.

Note per la battitura di SuperBasic

Per caricare SUPERBASIC è stato usato MLX (vedi programma precedente). Poiché SUPERBASIC è caricato nell'area di memoria normalmente occupata dal BASIC (previo un suo rilocamento in RAM), si è usato un'accorgimento particolare per evitare che SUPERBASIC ricopra una parte di MLX. La procedura, che vi

potrà tornare utile, la seguente:
1) Spegnere e riaccendere il C64;
2) Digitare le seguenti istruzioni e premere RETURN; POKE44,22; POKE642,22; POKE5632,0; NEW;
3) Caricare in memoria MLX e dare il RUN;
4) Alla richiesta degli indirizzi di partenza e di fine, fornire quelli del programma LM. (2049 e 5306 per il Superbasic);
5) Terminato il lavoro di battitura del SUPERBASIC, usare il SAVE dell'MLX per registrarlo. Fare attenzione che se non si termina il lavoro in una volta sola, occorre ripetere tutta la procedura descritta (punti 1-4), prima di riprendere la digitazione.
Una volta salvato l'intero SUPERBASIC su disco o nastro, potrete richiamarlo in memoria con un normale LOAD, e poi dare il RUN, senza più bisogno delle POKE prima descritte.

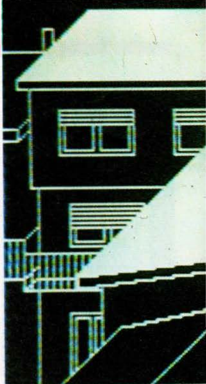
Utilizzo

Quando si fa partire il Superbasic, viene copiato il Basic dalla ROM nei corrispondenti indirizzi di RAM ($A000 - $BFFF), vengono poi modificati alcuni comandi e vengono aggiunte le routines in linguaggio macchina per la gestione di quelli nuovi. Queste routines vengono caricate dall'indirizzo $C000 all'indirizzo $CC00.
Il Superbasic è quindi incompatibile con i programmi che risiedono negli stessi utilizzi, può comunque coesistere con il DOS 5.1.
All'interno dei programmi che richiedono il Superbasic non può essere utilizzato il comando STOP che può essere sostituito dal comando END.
Tutti gli operandi dei comandi Superbasic possono essere espressi sia in valore assoluto che usando variabili impostate col valore desiderato.
Listando un programma contenente comandi Superbasic con lo stesso inattivo, il carattere "[" verrà listato come STOP.
Premendo RUN/STOP RESTORE si disabilita il Superbasic che può essere riattivato con il comando:
POKE 1,54.

COMANDI

Comandi standard modificati

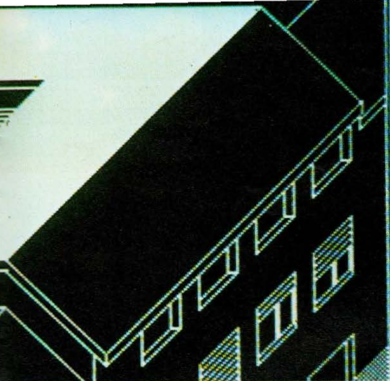
RESTORE: Può essere seguito da un numero di linea o da una variabile contenente un numero di linea. Questo fa si che una READ eseguita dopo il comando "RESTORE n" leggerà la linea di DATA di numero uguale o superiore ad "n".
LIST: il tasto SHIFT è in grado di interrompere la visualizzazione della lista di un programma sullo schermo finché è tenuto premuto.



GOTO
GOSUB
ON: Tutte le possibili forme di scrittura di questi comandi ammettono che il numero di linea sia indicato con una variabile.
Es.: 10 K = 1000
20 GOSUB K
ASC ($str): La funzione dà come risultato "0" se applicata ad una stringa nulla

Comandi per il controllo dei suoni

[SSND (Set-up SouND)
Il comando [SSND produce un suono e ne determina le caratteristiche.
Il volume del suono viene impostato al valore massimo.
[SSND voce, ad, sr, onda,



freq.[,pfreq]
voce Numero della voce usata. Può avere un valore compreso tra 1 e 3.
ad Valori di ATTACK e DECAY. Può assumere un valore tra 0 e 255.
sr Valori di SUSTAIN e RELEASE. Può assumere un valore tra 0 e 255.
onda Definisce la forma d'onda ovvero il timbro del suono. Possono essere definite quattro forme d'onda e per ogni forma d'onda 2 valori che fanno partire o fermare il suono.

freq Definisce la nota. Il suo valore può variare tra 0 e 65535.

FORMA D'ONDA	START	STOP
Triangolare	17	16
A dente di sega	33	32
Rettangolare (pulsante)	65	64
Rumore	129	128

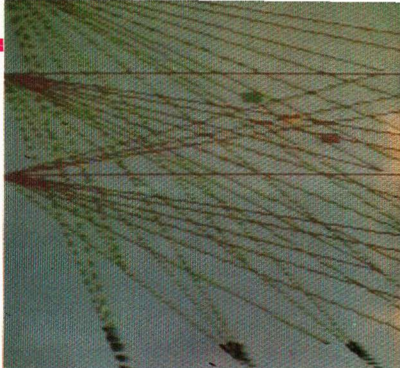
Far riferimento al "COMMODORE 64 PROGRAMMER'S REFENCE GUIDE", appendice E pag. 384 per il valore di ogni nota.
pfreq Va usato in congiunzione alla forma d'onda rettangolare (64/65). Definisce la lunghezza del ciclo di pulsazione ovvero la durata della parte positiva dell'onda. Può assumere un valore tra 0 e 12228.

Nota su AD/SR

Quando una nota viene emessa, passa da un volume 0 a un volume di picco in un certo tempo (controllato da ATTACK), per poi scendere ad un valore di regime (controllato da SUSTAIN) in un intervallo di tempo (controllato da DECAY) e si smorza infine a volume 0 in un certo tempo (controllato da RELEASE).
I valori di ATTACK, DECAY e RELEASE determinano una durata e variano tra 0 e 15. Vedi tabellina durate a pag. 198 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE".
SUSTAIN invece definisce il volume di mantenimento della nota rispetto al volume di picco. Varia tra 0 e 15 con scala lineare dove 0 significa "suono assente" e 15 "volume di picco".
Il valore AD è determinato dalla formula:
AD = ATTACK * 16 + DECAY
con formula analoga si determina SR
SR = SUSTAIN * 16 + RELEASE

[PLAY

Il comando [PLAY è una forma abbreviata di [SSND, va usato per modificare la nota, sempre che i valori AD/SR siano stati precedentemente impostati dalla [SSND.
[PLAY vconda, freq[, pfreq]
vconda Parametro che definisce sia la forma d'onda che la voce da usare. Fare riferimento agli operandi VOCE ed ONDA della [SSND per i valori ammessi.
Il contenuto di VCONDA è definito dalla seguente formula:
VCONDA = ONDA * 256 + VO-



CE
freq Definisce la nota. Il suo valore può variare tra 0 e 65535. Far riferimento al "Commodore 64 Programmer's Reference Guide", appendice E pag. 384 per i valori di ogni nota.
pfreq Va usato in congiunzione alla forma d'onda rettangolare (64/65). Definisce la lunghezza del ciclo di pulsazione ovvero la durata della parte positiva dell'onda. Può assumere un valore tra 0 e 12228.

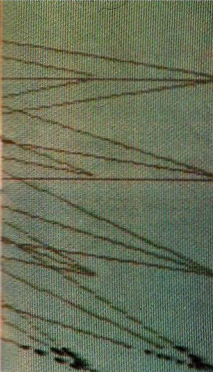
Comandi per il controllo dei colori

[BKGD (Back GrounD color)
Il comando [BKGD serve a definire il colore di fondo dello schermo.
[BKGD col
col Numero fra 0 e 15 che definisce il colore
[EXTC (EXTernal Color)
Il comando [EXTC serve a definire il colore del bordo dello schermo.
[EXTC col
col Numero fra 0 e 15 che definisce il colore
[FCOL (Fill COlor memory)
Il comando [FCOL riempie il buffer video della memoria colore con la tinta richiesta. Fa diventare dello stesso colore tutto lo schermo in high resolution o tutti i caratteri già presenti sullo schermo.
[FCOL col
col Numero fra 0 e 15 che definisce il colore

[SCRN (SCReeN colors set)
Il comando [SCRN serve a definire i colori di fondo, del bordo e dei caratteri in una sola operazione.
[SCRN bgc, boc, chc
bgc Numero fra 0 e 15 che definisce il colore di fondo dello schermo.
boc Numero fra 0 e 15 che definisce il colore del bordo dello schermo.
chc Numero fra 0 e 15 che definisce il colore dei caratteri sullo schermo.
[BKG4 (Back Ground colors)
Il comando [BKG4 definisce tutti i 4 colori di fondo dello schermo usabili in EXTENDED COLOR MODE e MULTICOLOR BIT MAP MODE.
[BKG4 col0, col1,col2,col3
Col0-3 Numeri fra 0 e 15 che definiscono rispettivamente il colore di fondo dello schermo (col0), il colore background 1 (col1), il colore background 2 (col2) e il colore dei caratteri (col3).
Per maggiori informazioni vedi i paragrafi MULTICOLOR MODE GRAPHICS ed EXTENDED BACKGROUND COLOR MODE da pag. 115 a 121 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE".

Comandi per il controllo degli sprites

[DSPR (Define SPRite)
Il comando [DSPR è usato per definire le caratteristiche di uno sprite.
[DSPR spr, blk, xexp, yexp, xpos, multi, col[, mc0, mc1]
sper Numero dello sprite definito, può variare tra 0 e 7
blk Sprite pointer. Può avere un valore tra 0 e 255.
Siccome per definire uno sprite occorre usare memoria, lo sprite pointer indica in quale blocco del banco da



16K visibile dal chip di gestione sprite VIC II risiede lo sprite stesso.
Per esempio, se il VIC II punta al banco 1 e blk = 14, lo sprite è definito alla posizione di memoria:
1 * 16384 + 14 * 64 = 17280
xexp Espansione orizzontale dello sprite.
0 = non espanso
1 = espanso
yexp Espansione verticale dello sprite.
0 = non espanso
1 = espanso
xpos Posizione iniziale dell'angolo superiore sinistro dello sprite sull'asse orizzontale.
Può assumere un valore da 0 a 512 (0 a sinistra)
Fare riferimento alla pag. 143 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE" per i valori che rendono visibile lo sprite sullo schermo.
ypos Posizione iniziale dell'angolo superiore sinistro dello sprite sull'asse verticale.
Può assumere un valore da 0 a 256 (0 in alto)
Fare riferimento alla pag. 143 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE" per i valori che rendono visibile lo sprite sullo schermo.
multi Definisce se lo sprite è multicolor oppure di un unico colore.
0 = colore singolo
1 = multicolor
Se multi = 1 devono essere definiti anche i parametri opzionali mc0 e mc1.
col Definisce il colore dello sprite o il colore base dello sprite se multicolor. Valore tra 0 e 15.
mc0 Definisce il secondo colore dello sprite multicolor. Valore tra 0 e 15.
mc1 Definisce il terzo colore dello sprite multicolor. Valore tra 0 e 15.

Nota sul colore degli sprites

Sprites di un solo colore:
I bits in ON dello sprite assumono il colore definito dal parapetro col.
1 bits in OFF hanno il colore di fondo dello schermo.
Sprites multicolor:
I bits degli sprites vengono considerati a coppie. Se il valore di una coppia è "00" questa assume il colore di fondo dello schermo.
Le coppie di valore "10" assumono il colore definito dal parametro col. Le coppie di valore "01" e "11" assumono rispettivamente i colori definiti nei parametri mc0 e mc1.

[ESPR (Enable SPRite)
Il comando [ESPR serve per abilitare lo sprite ovvero a renderlo visibile.
[ESPR spr
spr Numero dello sprite che si vuole abilitare. Può variare tra 0 e 7.

[KSPR (kill SPRite) il comando [KSPR serve per disabilitare lo sprite ovvero a renderlo invisibile.
[KSPR spr
spr Numero dello sprite che si vuole disabilitare. Può variare tra 0 e 7.

[BSPP (Background/SPrite Priority)
Il comando [BSPP serve a definire la priorità di uno sprite rispetto al "fondo", ovvero definisce se uno sprite passerà "dietro" ai caratteri dello schermo oppure "davanti".
[BSPP spr, sel
spr Numero tra 0 e 7 che definisce lo sprite.
sel Priorità dello sprite rispetto al background.
0 = lo sprite ha priorità maggiore e nasconde i caratteri
1 = lo sprite ha priorità più bassa e ne è nascosto

[MOVE (MOVE sprite)
Il comando [MOVE serve a muovere lo sprite sullo schermo.
[MOVE spr, xpos, ypos
spr Numero tra 0 e 7 che definisce lo sprite.
xpos Posizione dell'angolo superiore sinistro dello sprite sull'asse orizzontale.

Può assumere un valore da 0 a 512 (0 a sinistra)
Fare riferimento alla pag. 143 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE" per i valori che rendono visibile lo sprite sullo schermo.
ypos Posizione dell'angolo superiore sinistro dello sprite sull'asse verticale.
Può assumere un valore da 0 a 256 (0 in alto)
Fare riferimento alla pag. 143 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE" per i valori che rendono visibile lo sprite sullo schermo.

Comandi per il controllo del chip VIC II

Il chip VIC II può vedere solo 16K di memoria per volta, la memoria è perciò divisa in 4 banchi di 16K ciascuno.
Normalmente il chip accede ai primi 16K, ovvero al banco "0".
All'interno del banco selezionato, può essere definito quale blocco da 1K costituisce la memoria dello schermo (buffer video), normalmente il blocco "1".
Il chip VIC II deve inoltre selezionare il blocco da 2K che definisce il CHARACTER SET disponibile.
[BANK (define BANK)
Il comando [BANK definisce quale banco di memoria è visibile al VIC II.
[BANK sel
sel Numero che indica il blocco. Può variare tra 0 e 3
[VS1K (video Screen 1K)
Il comando [VSK 1 determina quale dei 16 blocchi da 1K, disponibili all'interno di un banco, viene usato per il buffer video.
US1K sel
sel Numero che indica il blocco. Può variare tra 0 e 15.
[CB2K (Character Block 2K)
Il comando [CB2K seleziona quale degli 8 blocchi da 2K, disponibili all'interno di un banco, viene usato per il character set.
[CB2K sel
sel Numero che indica il blocco. Può variare tra 0 e 7.
NOTA: [CB2K è usato anche per selezionare quale dei 2 blocchi da 8K disponibili è usato per la BIT MAP SCREEN memory.
I valori di SEL da 0 a 3 selezionano il primo blocco mentre i valori da 4 a 7 selezionano il blocco superiore.
I 3 comandi [BANK, [VS1K e [CB2K vanno usati insieme e in modo coordinato, facendo attenzione nel selezionare i blocchi usati. Il sistema può "cadere" se si selezionasse (ad esempio) per la memoria video la pagina "0" ($0000-$0400). Anche i banchi "2" e "3" devono essere utilizzati con cautela.
Spostando la memoria video, si può abbassare l'entry point dei programmi Basic da $0800 e $0400 guadagnando 1K.

Comandi per l'uso del BANCO 3

Quando viene selezionato il banco 3, il VIC II vede la RAM dall'indirizzo $C000 all'indirizzo $FFFF, ignorando la ROM che è però indispensabile al Basic.
Il Superbasic permette l'uso del banco 3 per allocarvi la memoria video ("VS1K 3" dall'indirizzo $CC00 che è quello successivo al massimo indirizzo occupato dal Superbasic). La RAM da $D000 può essere usata per character set, definizione sprites, bit map screen memory, etc. Per usare questi indirizzi bisogna abilitare, usare e disabilitare la RAM.
[HRAM (High RAM usage)
Il comando [HRAM permette che altri comandi Superbasic possano essere eseguiti usando il banco 3 di memoria.
[HRAM subcom [param]
subcom È un qualsiasi comando Superbasic ad eccezione del comando [MXGR.
param È la lista dei parametri richiesti dal comando "subcom"
[STUF (STore in core)
Il comando [STUF è un equivalente della POKE e funziona esattamente allo stesso modo. A differenza della POKE, può essere eseguito nel banco 3 come "subcom" del comando [HRAM.
[STUF loc, val
[HRAMSTUF loc, val
loc È la posizione di memo-

ria dove viene messo il dato "vai"
val È il dato da mettere nella posizione di memoria "loc".
[LOOK (LOOK in core)
Il comando [LOOK è un equivalente della PEEK; ha però una sintassi diversa perché richiede l'uso di una variabile dove mettere il valore estratto dalla memoria. Può essere eseguito nel banco 3 come "subcom" del comando [HRAM.
[LOOK loc, val
[HRAM LOOK loc, val
loc È la posizione di memoria da dove viene prelevato il dato "val"
Val È la variabile nella quale viene messo il dato prelevato dalla memoria "loc".

Comandi per il controllo della grafica

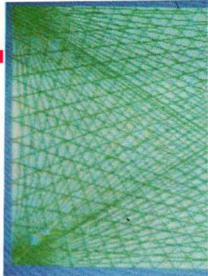
[ECGR (Extended Color mode GRaphic)
Il comando [ECGR seleziona la gestione della grafica in extended color mode.
[ECGR switch
Switch Abilitazione/disabilitazione extended color mode
0 = disabilitato
1 = abilitato
[MCGR (Multi Color mode GRaphic)
Il comando [MCGR seleziona la gestione della grafica in multi color mode.
[MCGR switch
switch Abilitazione/Disabilitazione multi color mode
0 = disabilitato
1 = abilitato
BMGR (bit Map mode GRaphic)
Il comando [BMGR seleziona la gestione della grafica in bit map mode.
[BMGR switch
switch abilitazione/disabilitazione bit map mode
0 = disabilitato
1 = abilitato
NOTA: [BMGR E [MCGR possono essere usati insieme per ottenere un bit map graphics in multi color mode.
[ECGR e [BMGR, se usati insieme, mettono a blank lo schermo velocizzando l'esecuzione dei programmi.
[DLCS (DownLoad Character Set)
Il comando [DLCS serve a copiare in RAM il character set.
Possono essere copiati i caratteri minuscoli, maiuscoli o entrambi.
[DLCS set, addr
set Indica il character set da copiare:
0 = set maiuscolo
1 = set minuscolo
2 = entrambi i sets
addr È l'indirizzo di partenza del nuovo character set.
Deve essere allineato ai 2K, a meno che non si desideri cambiare l'ordine di definizione dei caratteri all'interno del set.
Usando addr = 53248 il set sarà copiato nella RAM sottostante il character set standard in ROM nel BANCO 3.
Potrà essere modificato con l'uso della [HRAMSTUF.
[FBMS [Fill Bip Map Screen)
Il comando [FBMS va usato in BIT MAP MODE. Riempie il buffer video high resolution con la configurazione di bits prescelta.
[FBMS byte
byte È una variabile contenente un valore numerico tra 0 e 255 o il numero stesso.
[FSCR (Fill SCReen)
Il comando [FSCR riempie il buffer video con il carattere specificato. Può essere usato anche in BIT MAP MODE dove la memoria video serve a determinare il colore dello schermo high resolution.
[FSCR byte
byte È una variabile contenente la rappresentazione ASCII del carattere da mettere sul video o che definisce il colore da usare, oppure il numero stesso.
PLOT (PLOT pixel)
Il comando [PLOT va usato in BIT MAPE MODE, rende visibile uno dei 320 * 200 punti dello schermo.
[PLOT xpos, ypos
xpos Posizione del punto sull'asse orizzontale. Può avere un valore tra 0 e 319.
ypos Posizione del punto sull'asse verticale. Può avere un valore tra 0 e 199, dove "0" è il primo "raster" (riga scandita dal pennello elettronico) in alto.
[CLPX (CLear PiXel)
Il comando [CLPX va usato

in BIT MAPE MODE, cancella uno dei 320 * 200 punti dello schermo.
[CLPX xpos, ypos
xpos Posizione del punto sull'asse orizzontale. Può avere un valore tra 0 e 319.
ypos Posizione del punto sull'asse verticale. Può avere un valore tra 0 e 199, dove "0" è il primo "raster" (riga scandita dal pennello elettronico) in alto.

[FLIP (FLIP flop pixel)
Il comando [FLIP va usato in BIT MAP MODE, cambia lo stato di uno dei 320 * 200 punti dello schermo ovvero se è visibile lo rende invisibile e viceversa.
[FLIP xpos, ypos
xpos Posizione del punto sul'asse orizzontale. Può avere un valore tra 0 e 319.
ypos Posizione del punto sull'asse verticale. Può avere un valore tra 0 e 199, dove "0" è il primo "raster" (riga scandita dal pennello elettronico) in alto.

[MCPL Multi Color PLot pixel
Il comando [MCPL va usato in multicolor BIT MAP MODE, e definisce il colore e la posizione di uno dei 160 * 200 punti dello schermo.
[MCPL xpos, ypos, sel
xpos Posizione del punto sull'asse orizzontale. Può avere un valore tra 0 e 159.
ypos Posizione del punto sull'asse verticale. Può avere un valore tra 0 e 199, dove "0" è il primo "raster" (riga scandita dal pennello elettronico) in alto.
sel Può avere un valore tra 0 e 3 con i seguenti significati:
0 = colore del background screen
1 = colore definito nel semibyte di destra del buffer video
2 = colore definito nel semibyte di sinistra del buffer video
3 = colore definito nel buffer colore video.
Esempio: il buffer video è stato riempito con
[FSCR 16
ovvero - semibyte di sinistra = 1 = bianco
- semibyte di destra = 0 = nero.
Il buffer colore video è stato riempito di colore blu con
[FCOL 6
Il colore di sfondo dello schermo è stato impostato a giallo con
[BKGD7
Il comando
– [MCPL 30,40,0 genera un pixel giallo (invisibile)
– [MCPL 40,50,1 genera un pixel nero ... 0000
– [MCPL 50,60,2 genera un pixel bianco 0001 ...
– [MCPL 70,80,3 genera un pixel blu

[DRAW (DRAW linea)
Il comando [DRAW è usato in BIT MAP MODE (anche multicolor) per disegnare o cancellare un segmento sullo schermo.
[DRAW type, xpos1, ypos1, xpos2, ypos2
type Opzione di scrittura o di cancellazione
0 = cancellazione



1 = scrittura
xpos 1 Posizione del punto di inizio del segmento sull'asse orizzontale. Può avere un valore tra 0 e 319.
ypos 1 Posizione del punto di inizio del segmento sull'asse verticale. Può avere un valore tra 0 e 199.
xpos 2 Posizione del punto di fine del segmento sull'asse orizzontale. Può avere un valore tra 0 e 319.
ypos2 Posizione del punto di fine del segmento sull'asse verticale. Può avere un valore tra 0 e 199.

[HRCS (High Resolution Character Set)
Il comando [HRCS memorizza l'indirizzo di inizio del character set da usare in BIT MAP MODE.
Il character set può anche non essere allineato ai 2K e può essere il character set standard, in quest'ultimo caso l'indirizzo deve essere 53248 che seleziona il character set MAIUSCOLO in ROM.
[HRCS addr
addr Indirizzo di inizio del character set

[CODE (CODE character in screen code)
Il comando [CODE permette



di tradurre una stringa dalla codifica ASCII a quella SCREEN DISPLAY CODE, che è la codifica usata dai comandi [CHAR e [CHRX per scrivere dei caratteri sullo schermo in bit map mode.

[CODE str$

str$ È il nome di una variabile che contiene la stringa da convertire. Dopo l'esecuzione il contenuto della stringa è in SCREEN CODE e non più in ASCII; va quindi salvata prima se si vuole riutilizzarla.

I caratteri di controllo RVS-ON e RVS-OFF possono essere usati nella stringa per selezionare il set maiuscolo (OFF) o il set minuscolo (ON). Ogni altro carattere di controllo non deve essere usato nella stringa perché può dar luogo a risultati imprevedibili.

[CHAR (write CHARacter in bit map mode)

Il comando [CHAR permette di scrivere un carattere sullo schermo in bit map mode.

[CHAR byte, xpos, ypos

byte È la rappresentazione in screen code del carattere da mettere sullo schermo, come specifico nell'appendice B di pag. 376 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE". La variabile byte può essere ottenuta facendo, una dopo l'altra, le necessarie [CODE delle MID$ sulla stringa origine.

xpos È la posizione sull'asse orizzontale dell'angolo superiore sinistro della matrice carattere.

ypos È la posizione sull'asse verticale dell'angolo superiore sinistro della matrice carattere.

[CHRX (write CHaRacter in bit map mode with eXclusive OR)

Il comando [CHRX permette di scrivere un carattere sullo schermo in bit map mode facendo l'OR esclusivo con quanto già esistente sullo schermo in quella posizione. Può essere utilizzato per cancellare un carattere già presente sullo schermo.

[CHRX byte, xpos, ypos

byte È la rappresentazione in screen code del carattere da mettere sullo schermo, come specificato nell'appendice B di pag. 376 del "COMMODORE 64 PROGRAMMER'S REFERENCE GUIDE". La variabile byte può essere ottenuta facendo, una dopo l'altra, le necessarie [CODE delle MID$ sulla stringa origine.

xpos È la posizione sull'asse orizzontale dell'angolo superiore sinistro della matrice carattere.

ypos È la posizione sull'asse verticale dell'angolo superiore della matrice carattere.

[MXGR (MIXed mode GRaphic set-up)

Il comando [MSGR crea una interrupt ruotine che consente di gestire contemporaneamente lo schermo in 2 modi grafici diversi.

Ogni volta che il "raster register" (il registro che conta le righe orizzontali pennellate sullo schermo) eguaglia uno dei valori indicati, viene chiamata la interupt routine che sostituisce il valore indicato in registro del chip VIC II.

Ricordare che la parte visibile dello schermo va dal raster 51 al raster 251.

[MXGR reg, mask, rast1, val1, rast2, val2

reg È uno dei registri del VIC II. Il registro è indicato dall'indirizzo all'interno del chip. Se l'indirizzo iniziale del chip è 53248 e l'indirizzo del registro che controlla il colore di fondo dello schermo è 53281, l'operando reg in questo caso è "33" (53281 – 53248).

mask Controlla quali bit del registro vengono modificati. Può avere un valore tra 0 e 255.

I bit del registro che vengono modificati sono quelli corrispondenti ai bit in OFF della maschera (mask).

Per esempio la maschera "240" (($F0) – (11110000)) permette la modifica del solo semibyte di destra del registro.

rast1 Primo valore di ra-

ster per il quale viene attivata l'interrupt routine.
val1 Valore che viene inserito nei bit del registro (reg) corrispondenti ai bit in off della maschera (mask) quando il raster register eguaglia il valore specificato in rast1.
rast2 Secondo valore di raster per il quale viene attivata l'interrupt routine.
val2 Valore che viene inserito nei bit del registro (reg) corrispondenti ai bit in off della maschera (mask) quando il raster register eguaglia il valore specifico in rast2.
[MXGR
33,240,152,10,252,0
farà si che il fondo dello schermo (reg = 33) risulti nero nella metà superiore (raster 151) e rosa nella metà inferiore (raster 152 al raster 252).
Nota: La temporizzazione dell'interrupt routine può essere imprecisa perché la interrupt routine standard [IRQ) è sempre attiva. Ciò può causare delle ondulazioni sulla linea di separazione fra i "modi" dello schermo. Per mantenere questa linea precisa occorre disabilitare la [RQ perdendo cosi il controllo sulla tastiera e limitando l'I/O al solo joystick.
[CMXV (Change MIXed graphic mode Values)
Il comando [CMXV permette di variare dinamicamente i valori da inserire nel registro mentre la interrupt routine creata dalla [MXGR è attiva
[CMXV val1, val2
val1 Valore che viene inserito nei bit del registro (reg) corrispondenti ai bit in off della maschera (mask) quando il raster register eguaglia il valore specifico in rast1.
val2 Valore che viene inserito nei bit del registro (reg) corrispondenti ai bit in off della maschera (mask) quando il raster register eguaglia il valore specifico in rast2.
[KMXG (Kill MIXed Graphic mode)
Il comando [KMXG disabilita l'interrupt routine creata dalla [MXGR lasciando il registro in uno stato indeterminato. Per forzare il valore voluto nel registro usare prima la [CMXV. Non ha operandi.
[KMXG
I due comandi che seguono utilizzano la possibilità dell'integrato VIC II di eseguire lo scrolling lento dello schermo video sia in orizzontale che in verticale.
Questo tipo di movimento viene effettuato spostando l'intero schermo lateralmente o verticalmente di un solo punto per volta. Lo scrolling lento ha però come requisito la riduzione dello schermo nella direzione del movimento.
Per fare uno scrolling verticale lo schermo visibile deve essere portato da 25 a 24 righe, e per uno scrolling orizzontale da 40 a 38 caratteri per riga.
[SIZE (Screen SIZE)
Il comando [SIZE permette di selezionare il numero di righe e/o colonne che si vogliono visibili sullo schermo.
[SIZE colsel, rowsel
colsl Seleziona il numero di colonne visibili sullo schermo.
0 = 38 colonne
1 = 40 colonne
rowsel Seleziona il numero di righe visibili sullo schermo.
0 = 24 righe
1 = 25 righe
[XYSC (X-Y Scroll)
Il comando [XYSC muove l'intero schermo orizzontalmente e/o verticalmente da 1 a 7 pixel per volta.
Normalmente viene utilizzato lo spostamento di 1 pixel alla volta per realizzare lo scroll lento.
[XYSC xval, yval
xval Può assumere un valore tra 0 e 7.
Variando ad esempio il valore tra 0 e 7 o tra 7 e 0 lo schermo può essere mosso di un pixel alla volta rispettivamente verso l'alto o verso il basso.
yval Può assumere un valore tra 0 e 7.
Variando ad esempio il valore tra 0 e 7 o tra 7 o 0 lo schermo può essere mosso di un pixel alla volta rispettivamente verso sinistra o verso destra.
Esiste inoltre il comando [CATA che lista sullo schermo tutti i nuovi comandi del SUPERBASIC.
[CATA (CATAlog of command)
[CATA
Non ha operandi.
■

pubblicato da SuperVic e C.64 n. 3 - 1984

Bombardiere



Sei in missione per distruggere degli obiettivi nemici al suolo. Ma attenzione! il loro territorio è fortemente difeso dalla contraerea. La tua missione continua finché non vieni abbattuto, ma prima che ciò accada devi colpire il maggior numero di obiettivi. Lasciar cadere le bombe al momento giusto è già abbastanza difficile, dovendo poi evitare le esplosioni delle cannonate, la missione diventa impegnativa.

HAI VOLATO TROPPO BASSO!!

BOMBE SGANCIATE = 40

OBIETTIVI COLPITI = 0

COLPITO DOPO 2 TENTATIVI

TEMPO DI VOLO = 74 SECONDI

BATTI SPAZIO PER UN ALTRO VOLO

Come giocare

Inizialmente ti viene richiesto a quale livello vuoi giocare. Più alto sarà il livello e più intenso sarà il fuoco della contraerea. A livello 1 hai moltissimo tempo per decidere il momento giusto per sganciare le bombe e per evitare le schegge delle esplosioni. A livello 10 ti si presenterà un'intensa barriera di fuoco e non sopravviverai a lungo se non sei un abile pilota.
Premi RETURN e sul video comparirà un cielo serale e udrai il ronzio dei motori del bombardiere che apparirà sul lato sinistro del video. Puoi renderti conto della tua velocità guardando passare gli alberi sotto di te. Nel cielo puoi vedere i

lampi della contraerea e le nuvole di fumo delle granate. Se finisci contro queste nuvole vieni abbattuto. Puoi cambiare quota usando i tasti funzione F5 e F7, posti sulla destra della tastiera. Il tasto F5 fa cabrare il bombardiere mentre il tasto F7 lo fa picchiare. Se sali alla sommità del cielo puoi evitare più facilmente la contraerea, ma è anche più difficile bombardare i bersagli. Se invece voli troppo basso ti andrai a schiantare contro gli alberi.

Quando sei all'inizio del volo sentirai una rapida seguenza di beeps che segnala la mancanza di obiettivi nella zona. Appena i beeps diventano più rari significa che ti stai avvicinando all'obiettivo; nello stesso momento appare nell'angolo, in basso a sinistra, il numero 9 che andrà decrementandosi via via ti avvicini al bersaglio. Quando starà per passare da 1 a zero il bersaglio comparirà sul lato destro dello schermo.

Per sganciare le bombe occorre premere il tasto F3. Sentirai il lamento della bomba che cade e vedrai il lampo sul suolo, proprio sotto il bombardiere, e ne sentirai l'esplosione. Se colpisci il bersaglio vedrai lampeggiare l'intero schermo.

Più alto si trova il bombardiere quando sgancia le bombe e più tempo queste impiegheranno a raggiungere il suolo. Quindi calcola

```
10 REM **BOMBARDIERE**
20 PRINT"{CLR}":T=0:U=1:Z=0:V=53248:G=10
   24:H=55296:B=0:F=0:Y=0
30 GB=G+852:HB=H+852:FF=80:GS=G+879:GT=G
   +961:GF=G+80:HF=H+80:BH%=0
40 FORJ=12288TO12323:READX:POKEJ,X:NEXT
50 FORJ=12324TO12350:POKEJ,0:NEXT
60 FORJ=12544TO12617:READX:POKEJ,X:NEXT
70 POKE53270,PEEK(53270)AND247:POKE649,1
   :POKE52,48:POKE56,48
80 S=54272:FORJ=0TO4:POKES+J,0:NEXT
90 Y=130:FB=0:BT=0:BD=0
100 TT=0:BG=0:A$=""
110 INPUT"{CLR}{WHITE}{2*CD}{2*CR}LIVELL
    O DI GIOCO (1-10)";L$
120 L=VAL(L$):IFL<1ORL>10THEN110
130 POKES+24,47:POKES,12:POKES+1,1:POKES
    +6,143:POKES+21,7:POKES+23,1:POKES+1
    7,1
140 POKES+4,129:POKES+14,88:POKES+15,115
    :POKES+20,240
150 PRINT"{CLR}":POKE53280,0:POKE53281,6
160 FORJ=0TO1023:POKEH+J,Z:NEXT:POKEH+96
    1,U
170 FORJ=1TO38:POKEG+880+J,102:NEXT
180 POKE2040,192:POKEV+39,0
190 POKEV,104:POKEV+1,Y:POKEV+21,1
200 TI$="000000":POKE650,128::L=L/20
210 E=RND(U):X=PEEK(V+31)
220 POKEHB,Z
230 IFE<LTHENF=RND(U)*760:POKEHF+F,7:POK
    EGF+F,42:GOSUB1000
240 SYS(12586)
250 IFE<.3THENPOKEGS,88
260 IFB=16THENPOKEGS,160
270 IFE>.95ANDFB=ZTHENB=101:FB=U:TT=TT+U
280 IFB/10=INT(B/10)THENPOKES+18,65:POKE
    GT,B/10+47
290 POKES+18,64:IFFB=UTHENB=B-U
300 IFB=ZTHENFB=Z
310 GETA$:IFA$=""THEN380
320 IFA$=CHR$(134)ANDBG=ZTHENBH%=210-Y:B
    G=U:BD=BD+U:POKES+20,248:GOTO370
330 IFA$=CHR$(135)THENY=Y-8
340 IFA$=CHR$(136)THENY=Y+8
350 IFY<ZTHENY=Z
360 POKEV+U,Y
370 A$=""
380 IFBG=UTHENPOKES+18,64:POKES+15,130+B
    H%/2:POKES+18,65
390 SYS(12544)
400 IFBG=ZTHENPOKES+18,64
410 IFPEEK(V+31)=UTHEN600
420 POKEH+F,0
430 BH%=BH%-4
440 IFB=2ANDBH%=ZTHENGOSUB2000
```

```
450 SYS(12586)
460 SYS(12544)
470 BH%=BH%-4
480 IFBG=UTHENPOKES+18,64:POKES+15,130+B
    H%/2:POKES+18,65
490 IFPEEK(V+31)=1THEN600
500 IFB=2ANDBH%=ZTHENGOSUB2000
510 IFBG=UANDBH%=ZTTHENPOKEHB,1:POKEGB,2
    14:POKES+18,64:BG=0:POKES+20,240:GOS
    UB1000
520 GOTO210
600 PRINT"{CLR}{WHITE}{2*CD}{3*CR}";:POK
    E53280,0:POKE53281,2:POKEV+21,0
610 FORJ=0TO24:POKES+J,0:NEXT
620 POKES+24,15:POKES,200:POKES+1,2:POKE
    S+5,13:POKES+6,248:POKES+4,129
630 FORK=1TO1500:NEXT
640 POKES+4,128
650 IFY>206THENPRINT"{RVS ON}HAI VOLATO
    TROPPO BASSO!!":GOTO670
660 PRINT"{RVS ON}SEI STATO ABBATTUTO!!"
670 PRINT"{2*CD}{3*CR}BOMBE SGANCIATE ="
    ;BD
680 PRINT"{2*CD}{3*CR}OBIETTIVI COLPITI
    =";BT
690 PRINT"{2*CD}{3*CR}COLPITO DOPO";TT;"
     TENTATIVI"
700 PRINT"{2*CD}{3*CR}TEMPO DI VOLO =";I
    NT(TI/60);" SECONDI"
710 PRINT"{5*CD}    BATTI SPAZIO PER UN
    ALTRO VOLO"
720 GET A$:IFA$<>" "THEN720
730 GOTO80
1000 POKES+7,25:POKES+8,1:POKES+12,15:PO
     KES+13,240
1010 POKES+11,129:POKES+11,128
1020 RETURN
2000 POKEHB,U:POKEGB,214:POKE53280,U
2010 GOSUB1000:GOSUB1000
2020 BT=BT+U
2030 POKEHB,Z:POKEGB,32:POKE53280,Z
2040 RETURN
3000 DATA0,4,0,0,6,0,0,7,0,128,7,128,192
     ,7,192,224,7,224
3010 DATA224,7,240,224,7,248,240,7,252,2
     55,255,255,255,255,255,0,248,12
4000 DATA169,0,133,251,169,4,133,252,160
     ,1,169,22,170,177,251
4010 DATA136,145,251,202,240,12,24,152,1
     05,41,168,144,241
4020 DATA230,252,76,13,49,230,251,169,40
     ,197,251,208,219,96
4030 DATA160,0,169,4,133,254,169,119,133
     ,253,162,20,169,32,145,253
4040 DATA202,208,1,96,24,152,105,40,168,
     144,241,230,254,76,54,49
```

bene i tempi di sganciamento! Se sei quasi in cima allo schermo, ed il numero in basso a sinistra sarà circa 2 o 1, dovrai sganciare premendo il tasto F3.
Presto o tardi verrai abbattuto. Quando ciò accade compariranno sul video i risultati della tua missione.

Tattiche vincenti

Volare alto è un modo per evitare la contraerea, ma è più difficile colpire il bersaglio. Un attacco a bassa quota rende le bombe più efficaci, ma un brusco movimento per evitare una scheggia può farti sbattere contro gli alberi.
Puoi sganciare bombe quando vuoi. Vale la pena fare un po' di pratica mirando anche agli alberi che sfilano al suolo. Puoi scoprire così a che distanza dovrebbe trovarsi l'obiettivo per venire colpito da ogni quota.
Quando sei diventato un abile pilota cerca di non sprecare bombe e di fare sempre centro sull'obiettivo.

Digitazione

Il programma comprende due routines in linguaggio macchina per il movimento degli alberi, degli obiettivi e delle schegge che sorpassano il bombardiere a velocità elevata. Tali routines sono caricate nella memo-

ria grazie alle istruzioni DATA delle linee 4000-4050. È essenziale digitare ogni numero esattamente, (anche un minimo errore può causare effetti stranissimi) oppure ricorrere al programma già registrato su cassetta.

La linea 510 è molto lunga. Usa l'abbreviazione per le istruzioni POKE (la lettera P e poi il tasto SHIFT CON LA LETTERA O).

I caratteri di controllo usati sono:

CLEAR: linee 20, 110, 600
CTRL-2: linee 110, 600
CRSR DOWN: linee 110(2), 600(2), 670(2), 680(2), 690(2), 700(2), 710(5)
CRSR RIGHT: linee 110(2), 600(3), 670(3), 680(3), 690(3), 700(3), 710(2).

Sezioni del programma

20-30 Inizializzazione della maggior parte delle variabili.
40-50 Formazione dello sprite.
60 Formazione della routine in linguaggio macchina.
70-80 Determinazione ampiezza del video e della memoria.
90-100 Inizializzazione dei video.
110-120 Richiesta del livello di gioco.
130-140 Assetto del chip del suono.
150-170 Costruzione dell'immagine.
180-190 Assetto dei registri dello sprite.
200 Assetto dell'orologio.
210-520 Ciclo principale.
600-660 Routine di scontro con effetti visivi.
670-700 Visualizzazione dei risultati della missione.
710-730 Invito a giocare ancora.
1000-1020 Subroutine per produrre il suono dell'esplosione.
2000-2040 Subroutine per visualizzare le bombe che esplodono accompagnate dagli effetti sonori.
3000-3010 Istruzioni DATA per lo sprite.
4000-4040 Istruzioni DATA per la routine in linguaggio macchina.

Punti di interesse

Il maggior punto d'interesse di questo gioco è la sua alta velocità. Una routine in linguaggio macchina si prende cura dello scorrimento dell'immagine verso sinistra; un'altra cancella la colonna sulla destra prima che sia fatta scorrere. Tale colonna è fuori visuale poiché il video, precedentemente, è stato ridotto a 38 colonne.

Per sveltire le operazioni del loop principale, che è in Basic, vengono sovente usate delle variabili alle quali sono assegnate delle costanti, per esempio: GB e HB sono le locazioni video del suolo perpendicolari al bombardiere. Altri esempi sono la variabile U, per l'unità (=1), e Z per zero. Tutto ciò per velocizzare le operazioni del microprocessore che non deve convertire ogni volta un numero decimale in binario.

Il manuale del Commodore 64 (programmer's Reference Guide) riporta una sezione sullo "Smooth Scrolling". Questo accorgimento non viene usato in questo programma perché è troppo lento e necessità di una routine in linguaggio macchina. La routine nelle linee 4000-4020 può essere usata per lo scorrimento dell'immagine da destra verso sinistra. In questo programma, si fanno scorrere 38 colonne video ad eccezione delle ultime tre righe dal basso. Per fare scorrere l'intera immagine bisogna modificare il numero 22 in 25 nella linea 4000.

Tratto dal
Libro dei Giochi
C. 64